Computers and Electrical Engineering 49 (2016) 84–94

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Optimized FPGA based continuous wavelet transform ☆

Yahya T. Qassim [a,*], Tim R.H. Cutmore [b], David D. Rowlands [a]

[a] School of Electronic Engineering, Griffith University, 170 Kessels Road, Nathan, Brisbane, QLD 4111, Australia
[b] School of Applied Psychology, Griffith University, Mt Gravatt Campus, Kessels Road, Brisbane, QLD 4111, Australia

A R T I C L E I N F O

Article history:
Received 24 December 2013
Received in revised form 27 November 2014
Accepted 27 November 2014
Available online 26 December 2014

Keywords:
CWT
EEG
ERP
FPGA
Optimization

A B S T R A C T

A memory efficient field programmable gate array (FPGA) method is described that facilitates the processing of the continuous wavelet transform (CWT) arithmetic operations. The CWT computations were performed in Fourier space and implemented on FPGA following several optimization schemes. First, the adapted wavelet function was stored in a lookup table instead of computing the equation each time. Second, the utilization of FPGA memory was highly optimized by only storing the nonzero values of the wavelet function. This reduces 89% of the memory storage and allows fitting the entire design into the FPGA. Third, the design decreases the number of multiplications and shortens the time to produce the CWT coefficients. The proposed design was tested using EEG data and demonstrated to be suitable for extracting features from the event related potentials. Fourth, wavelet function scales were eliminated which saves further resources. The achieved computation speed allows for real time CWT application.

1. Introduction

The one dimensional continuous wavelet transform (1-D CWT) is a widely used feature extraction tool for nonstationary signals with applications to many different disciplines [1–3]. The complexity implied in the CWT belongs to the high number of convolutions involved especially between large sequences. In case of using the CWT in real time applications, high processing speed becomes critical to overcome these heavy convolution calculations and this can be achieved by using the field programmable gate array (FPGA) platform [4]. There are only a few studies in the literature concerning the implementation of the CWT into VLSI [5–9]. The complexity of mapping the CWT convolver in VLSI design was investigated in [5] and various realizations were presented, although they all depend on the convolution method. Other works rely on a General Purpose Processor (GPP) or DSP processor [7,8] or require high end FPGA devices [6,9] to implement the CWT. What has not been published is a CWT design that is fast and can be implemented on low cost FPGA devices.

Mathematically, the CWT is the convolution between the analyzed signal $X(t)$ and the wavelet function $\psi(t)$ in the time domain such that [10]:

$$C(s,b) = \int_{-\infty}^{\infty} X(t).\psi^{*}_{s,b}(t).dt \tag{1}$$

where $C(s, b)$ is the wavelet coefficient at time b and scale s and the symbol $*$ refers to the complex conjugate.

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Associate Editor Dr. Rene Cumplido.
* Corresponding author. Tel.: +61 431255842.
E-mail addresses: yahya-taher.qassim@griffithuni.edu.au (Y.T. Qassim), t.cutmore@griffith.edu.au (T.R.H. Cutmore), d.rowlands@griffith.edu.au (D.D. Rowlands).

http://dx.doi.org/10.1016/j.compeleceng.2014.11.012
0045-7906/© 2014 Elsevier Ltd. All rights reserved.

Eq. (1) can be implemented in the time domain for small size of input signals where the number of total convolutions is also small.

An alternative method to calculate the CWT for a sampled signal is to use Fourier space instead of the time domain [11]. This can be achieved by transforming both input signals of (1) $X(t)$ and $\psi_{s,b}(t)$ into the frequency domain using the fast Fourier transform (FFT) which replaces the complex convolution with simple multiplication using the following relationship [12]:

$$g1(t) * g2(t) \Longleftrightarrow G1(\omega) \times G2(\omega) \tag{2}$$

where lowercase g is the time domain components and uppercase G is the relative frequency domain representation. The product can be transformed back to the time domain using the inverse FFT giving the CWT coefficients [11]. Some of the most commonly used nonorthogonal wavelet functions in the CWT analysis are the Morlet, Paul and the Mexican hat. Their formulas in the time and frequency domain are shown in Table 1 [11] where the representation of these formulas in both time and frequency domains are computationally complex. For example, in the time domain, the Morlet wavelet function consists of a complex sinusoid in the term $e^{i\omega_0 t}$ multiplied by a Gaussian envelope. The spectrum of this modulated Gaussian permits for simple interpretation of results due to its smoothness [1]. The normalization factor $\pi^{-1/4}$ ensures that the Morlet wavelet has unit energy. In the frequency domain, the Morlet wavelet uses the Heaviside step function and in both domains, the Morlet expressions are complex. The mathematical background for the Paul and the DOG wavelet functions are also complex in both domains as one can see from Table 1.

The selection of the more appropriate wavelet function for a given application depends on the information required to be extracted from the signal. For example, detecting evolutionary or transient phenomena in a signal requires a wavelet function that reflects more localized wavelet coefficients [13].

Compared against the previous works presented above, the CWT design in this paper is a novel, generalized and configurable feature extraction engine for low end FPGA platforms. The design uses Fourier space techniques and employs several optimization methods to improve speed and significantly reduce resource requirements. The proposed design is not limited to a fixed wavelet function $\psi(t)$ but can be easily adapted to different wavelet functions without the need to resynthesize or redesign the circuit. The proposed design uses optimized lookup tables (LUTs) in a block RAM (BRAM) to store the pre-calculated wavelet function. This is advantageous since the pre-calculated wavelet function only needs to be downloaded to the BRAM which means that the circuit design is not affected. It also means that the contents of the LUT can be easily replaced by another wavelet function when required by changing the LUT contents. This makes the low cost Spartan 3AN (1.4 M gate) [4] suitable for the proposed design.

This paper is organized as follows; Section 2 presents the CWT design description, Section 3 gives the optimizations used in the CWT design and Section 4 outlines an implementation example based on Electroencephalogram (EEG) analysis. Section 5 presents the discussion and conclusions are provided in Section 6.

2. The proposed design

The computation of the CWT in Fourier space has been previously shown as an efficient and quick approach to implement the CWT using the FFT [12]. The design flow for the FFT based digital CWT can be seen in Fig. 1 [14]. From Fig. 1, block A contains the wavelet function in the time domain, $g2(t)$, at different scales and the FFT process required to transfer this function to the frequency domain. Both the input signal, $g1(t)$, of length 2^n, and the wavelet function ($g2(t)$) are stored in a buffer. After applying the FFT on both $g1(t)$ and $g2(t)$ the results $G1(\omega)$ and $G2(\omega)$ are stored in buffers $C1$ and $C2$ respectively. The contents of these two buffers are multiplied before applying an IFFT to produce the wavelet coefficients at all the wavelet scales.

Closer examination of the design flow for the CWT in Fig. 1 reveals that the wavelet functions in block A can be implemented using 3 possible methods:

Table 1
Three different wavelet bases in two domains [11].

Function	Time domain (t)	Frequency domain (ω)
Morlet (ω_0 = frequency)	$\pi^{-1/4} e^{i\omega_0 t} e^{-t^2/2}$	$\pi^{-1/4} H(\omega) e^{-(s\omega-\omega_0)^2/2}$
Paul (m = order)	$\frac{2^m i^m m!}{\sqrt{\pi(2m)!}}(1-it)^{-(m+1)}$	$\frac{2^m}{\sqrt{m(2m-1)!}} H(\omega)(s\omega)^m e^{-s\omega}$
Mexican hat (m = derivative)	$\frac{(-1)^{m+1}}{\sqrt{\Gamma(m+\frac{1}{2})}} \frac{d^m}{dt^m}\left(e^{-\frac{t^2}{2}}\right)$	$\frac{i^m}{\sqrt{\Gamma(m+\frac{1}{2})}}(s\omega)^m e^{-(s\omega)^2/2}$

$H(\omega)$ is the Heaviside step function (for Morlet and Paul), $H(\omega) = 1$ if $\omega > 0$, $H(\omega) = 0$ elsewhere, ω_0 is the nondimensional frequency, s is the scale and $\pi^{-1/4}$ is the normalization factor.
Γ stands for the gamma function.